CLAIMS

1. A method for analyzing a computer program, comprising the steps of:

defining a plurality of breakpoints for said computer program, at least one of said breakpoints including a respective expected code path condition;

executing said computer program;

with respect to each breakpoint including a respective expected code path condition encountered during execution of said computer program, automatically determining whether an actual code path taken during execution of said computer program matches the respective expected code path condition of the encountered breakpoint; and

automatically halting execution of said program if said actual code path taken during execution of said computer program does not match the expected code path condition.

2. The method for analyzing a computer program of claim 1, wherein said step of defining a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, defining a respective set of expected predecessor breakpoints.

3. The method for analyzing a computer program of claim 1, wherein said step of defining a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, defining a respective set of basic blocks required to be encountered during execution before the respective breakpoint.

4. The method for analyzing a computer program of claim 1, wherein said step of defining a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, defining a respective set of basic blocks prohibited from being encountered during execution before the respective breakpoint.

5. The method for analyzing a computer program of claim 1, wherein said step of defining a plurality of breakpoints comprises, for at least one of said breakpoints including a respective expected code path condition, at least one respective additional condition.

6. The method for analyzing a computer program of claim 5, wherein said step of halting execution of said program is performed if said actual code path taken during execution of said computer program does not match the expected code path condition, or if said at least one respective additional condition is met.

7. The method for analyzing a computer program of claim 1, wherein said step of defining a plurality of breakpoints comprises the steps of:

displaying code statements of said computer program to a user;

interactively receiving a user selection of a code statement as the location of a breakpoint having an expected code path condition;

responsive to receiving said user selection, displaying code statements which necessarily execute before the selected code statement in a highlighted form; and

interactively receiving a user selection of said expected code path condition.

8. The method for analyzing a computer program of claim 7, wherein said step of interactively receiving a user selection of said expected code path condition comprises:

interactively receiving a user selection of a code statement as the location of a basic block of code with which an expectation with respect to code path during execution is associated;

responsive to receiving said user selection of a code statement as the location of a basic block, determining the basic block to which the selected code statement belongs;

displaying code statements in the basic block containing said selected code statement in a highlighted form.

9. The method for analyzing a computer program of claim 8, further comprising:

interactively receiving a user indication of said expectation with respect to code path during execution associated with the basic block containing the selected code statement;

wherein said step of displaying code statements in the basic block containing the selected code statement in a highlighted form comprises displaying said statements in a first highlighted form responsive to a user indication of a first expectation with respect to code path during execution, and displaying said statements in a second highlighted form responsive to a user indication of a second expectation with respect to code path during execution, said second highlighted form being visually distinct from said first highlighted form.

10. The method for analyzing a computer program of claim 1,

wherein said step of defining a plurality of breakpoints comprises generating a respective breakpoint definition record for each of said plurality of breakpoints, said breakpoint definition record being separate from executable code of said computer program; and

wherein said step of executing said computer program includes monitoring said program during execution to determine whether a breakpoint has been encountered, and invoking a breakpoint handler responsive to encountering a breakpoint during execution, said breakpoint handler performing said step of determining whether an actual code path taken during execution of said computer program matches the respective expected code path condition of the encountered breakpoint.

11. A computer program product for analyzing a developing computer program, comprising:

a plurality of executable instructions recorded on signal-bearing media, wherein said instructions, when executed by at least one processor of a digital computing device, cause the device to perform the steps of:

receiving a user-specified definition of a plurality of breakpoints for said developing computer program, at least one of said breakpoints including a respective expected code path condition;

monitoring said plurality of breakpoints during execution of said developing computer program;

with respect to each breakpoint including a respective expected code path condition encountered during execution of said developing computer program, determining whether an actual code path taken during execution of said developing computer program matches the respective expected code path condition of the encountered breakpoint; and

halting execution of said developing computer program if said actual code path taken during execution of said developing computer program does not match the expected code path condition.

12. The program product of claim 11, wherein said program product is an integrated programming development environment comprising a source editor, a compiler, and a debugger.

13. The program product of claim 11, wherein said step of receiving a user-specified definition of a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, receiving a user-specified definition of a respective set of expected predecessor breakpoints.

14. The program product of claim 11, wherein said step of receiving a user-specified definition of a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, receiving a user-specified definition of a respective set of basic blocks required to be encountered during execution before the respective breakpoint.

15. The program product of claim 11, wherein said step of receiving a user-specified definition of a plurality of breakpoints comprises, for said at least one breakpoint including a respective expected code path condition, receiving a user-specified definition of a respective set of basic blocks prohibited from being encountered during execution before the respective breakpoint.

16. The program product of claim 11, wherein said step of receiving a user-specified definition of a plurality of breakpoints comprises, for at least one of said breakpoints including a respective expected code path condition, receiving a user-specified definition of at least one respective additional condition.

17. The program product of claim 16, wherein said step of halting execution of said program is performed if said actual code path taken during execution of said computer program does not match the expected code path condition, or if said at least one respective additional condition is met.

18. The program product of claim 11, wherein said step of receiving a user-specified definition of a plurality of breakpoints comprises the steps of:

displaying code statements of said computer program to a user;

interactively receiving a user selection of a code statement as the location of a breakpoint having an expected code path condition;

responsive to receiving said user selection, displaying code statements which necessarily execute before the selected code statement in a highlighted form; and

interactively receiving a user selection of said expected code path condition.

19. The program product of claim 18, wherein said step of interactively receiving a user selection of said expected code path condition comprises:

interactively receiving a user selection of a code statement as the location of a basic block of code with which an expectation with respect to code path during execution is associated;

responsive to receiving said user selection of a code statement as the location of a basic block, determining the basic block to which the selected code statement belongs;

displaying code statements in the basic block containing said selected code statement in a highlighted form.

20. The program product of claim 19, wherein said program product further causes the digital computing device to perform the step of:

interactively receiving a user indication of said expectation with respect to code path during execution associated with the basic block containing the selected code statement;

wherein said step of displaying code statements in the basic block containing the selected code statement in a highlighted form comprises displaying said statements in a first highlighted form responsive to a user indication of a first expectation with respect to code path during execution, and displaying said statements in a second highlighted form responsive to a user indication of a second expectation with respect to code path during execution, said second highlighted form being visually distinct from said first highlighted form.